El Proceso Software

Índice

Proceso Software

Ventajas de definir un Proceso Software

Un proyecto sin estructura es un proyecto inmanejable: no puede ser planificado, ni estimado, ni su progreso ser controlado, y mucho menos alcanzar un determinado compromiso de costos o tiempos. La idea originaria de buscar ciclos de vida que describan los estados por los que pasa el producto o procesos software que describan las actividades a realizar para transformar el producto, surge de la necesidad de tener un esquema que sirva como base para planificar, organizar, asignar personal, coordinar, presupuestar y dirigir las actividades de la construcción de software. De hecho, muchos proyectos han terminado mal porque las fases de desarrollo se realizaron en un orden erróneo.

Es por ello que al comienzo de un proyecto software, se debe elegir el ciclo de vida que seguirá el producto a construir, en base a las consideraciones antes mencionadas. El modelo de ciclo de vida elegido llevará a encadenar las tareas y actividades del proceso software de una determinada manera: algunas tareas no serán realizadas, otras deberán realizarse más de una vez, etc. Una vez conseguido un proceso software concreto para el proyecto, asignar personas a las distintas tareas, presupuestar los costos del proyecto, etc.

Concretamente, los procesos software se usan como:

- Guía normativa sobre la documentación a producir para enviar al cliente
- Base para determinar qué herramientas, técnicas y metodologías de ingeniería de software será más apropiadas para soportar las diferentes actividades
- Marco para analizar o estimar patrones de asignación y consumo de recursos a los largo de todo el ciclo de vida del producto software.
- Base para llevar a cabo estudios empíricos para determinar qué afecta a la productividad, el costo y la calidad del software.
- Descripción comparativa sobre cómo los sistemas software llegan a ser lo que son.

Estándar IEEE sobre Proceso Software

Describiremos a continuación las fases o subprocesos que conforman el proceso base de construcción del software y que se corresponden con el estándar IEEE.

Para cada subproceso se detalla propósito, actividades involucradas y documentación principal propuesta por el estándar. El estándar IEEE 1074-1989 determina el conjunto de actividades esenciales, no ordenadas en el tiempo, que deben ser incorporadas dentro de un modelo de ciclo de vida del producto software. Este modelo es seleccionado y establecido por el usuario para el proyecto a desarrollar, ya que la norma no define un ciclo de vida en particular.

El estándar, además incluye:

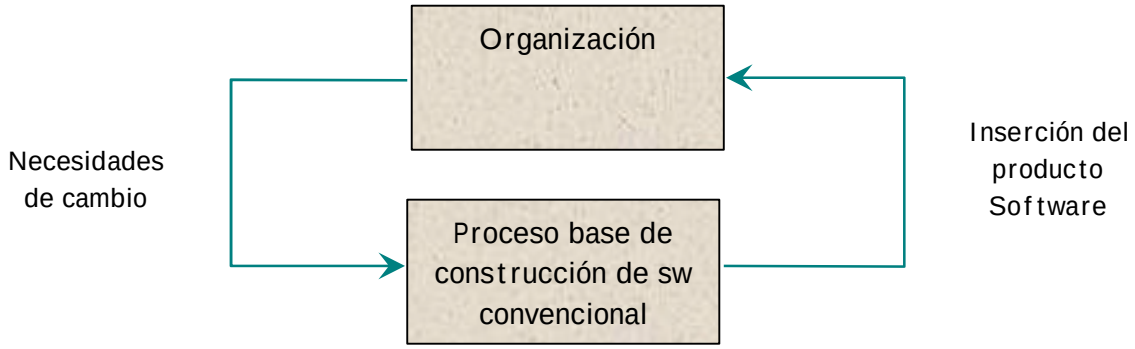
- Descripción de cada actividad
- Informe de entrada y salida para cada actividad
- Fuente y destino de la información a nivel de proceso y de actividad, que refleja la relación entre los procesos
- Productos obtenidos por el desarrollo de cada actividad

Este estándar fue escrito por organizaciones responsables de la gestión y desarrollo del software. Está dirigido a los gestores de proyectos, desarrolladores de software, responsables de la garantía de calidad, a quienes ejecutan tareas de apoyo, usuarios y personal que realiza mantenimiento del software.

En este texto simplemente se describirá el proceso software global, de modo de proporcionar una visión general.

Descripción global del proceso

El proceso base de construcción de software consiste en analizar las necesidades de la organización en un dominio, desarrollar una solución que las satisfaga y posteriormente reinsertar la solución en el dominio, bajo un marco de gestión, seguimiento, control y gestión de calidad.



El proceso software está compuesto de cuatro procesos principales, cada uno de los cuales agrupa una serie de actividades que se encargan de la realización de sus requisitos asociados:

Proceso de selección de un modelo de ciclo de vida del producto: identifica y selecciona un ciclo de vida para el software que se va a construir.

Proceso de gestión del proyecto: crean la estructura del proyecto y aseguran el nivel apropiado de la gestión del mismo durante todo el ciclo de vida del software.

Procesos orientados al desarrollo del software: producen, instalan, operan y mantienen el software y lo retiran de su uso. Se clasifican en procesos de pre-desarrollo, desarrollo y post-desarrollo.

Procesos integrales del proyecto: son necesarios para completar con éxito las actividades del proyecto software. Aseguran la terminación y calidad de las funciones del

mismo. Son simultáneos a los procesos orientados al desarrollo del software e incluyen actividades de no desarrollo.

Proceso de selección de un modelo de ciclo de vida del software

Ya dijimos que la selección de un ciclo de vida implica el establecimiento de un orden de ejecución de las distintas actividades, marcadas por el proceso e involucradas en el proyecto.

Sobre la base del tipo de producto software a desarrollar y a los requisitos del proyecto se identifican y analizan posibles modelos de ciclo de vida para dicho proyecto y se selección un único modelo que soporte adecuadamente el proyecto.

El estándar no dictamina ni define un ciclo de vida del software específico, ni una metodología de desarrollo, sólo requiere elegir y seguir un modelo de ciclo de vida.

La información de salida de este proceso es el **modelo de ciclo de vida del software** seleccionado.

Actividades a realizar:

Identificar los posibles modelos de ciclo de vida

Seleccionar el modelo más adecuado para el proyecto

Documentos de salida

Modelo de ciclo de vida seleccionado. (esto incluye la justificación de la elección que debe quedar documentada).

Proceso de gestión del proyecto

La gestión del proyecto presupone establecer condiciones para el desarrollo del mismo. Involucra actividades de planificación, estimación de recursos, seguimiento y control y evaluación del proyecto.

La **planificación** de proyectos se define como la predicción del a duración de las actividades y tareas a nivel individual, los recursos requeridos, la concurrencia y solapamiento de tareas para ser desarrollados en paralelo y el camino crítico a través de la red de actividades.

La **estimación** se define como la predicción de personal, esfuerzo y costos que se requerirá para terminar todas las actividades y productos conocidos, asociados con el proyecto.

La determinación del tamaño del producto a desarrollar es una de las primeras tareas en la gestión del proyecto, ya que sin unos conocimientos razonables, es imposible planificar o estimar el esfuerzo involucrado. El tamaño se define como la cantidad de código fuente, especificaciones, casos de prueba, documentación del usuario y otros productos tangibles que son salida del proyecto. La determinación del tamaño se basa principalmente en la experiencia de proyectos anteriores.

El seguimiento de proyectos es la recolección de datos y su acumulación sobre recursos consumidos, costes generados, e hitos asociados al proyecto. **Medir** en un proyecto se define como el registro de todos los productos generados en el mismo, de todos los recursos requeridos, planificación y solapamiento de todas las actividades y tareas y de todos los factores que impactan en el proyecto (conocimientos, métodos, herramientas, lenguajes, limitaciones, problemas y el entorno físico). La medición en los proyectos desarrollo de software es una actividad fundamental para la mejora de la productividad, el costo y la calidad del producto final. La medición, entrada a estudios empíricos, es el único modo para detectar fallos en el proceso de construcción de software.

Proceso de iniciación del proyecto

Abarca aquellas actividades de creación de la estructura del proyecto. Durante este proceso se define el ciclo de vida del software (asignación de responsables de cada actividad), para este proyecto y se establecen los planes para su gestión. Se estiman y se asignan los recursos; esto consiste en determinar los costos y recursos necesarios a fin de ejecutar las distintas tareas que demanda el proyecto. Se identifican y seleccionan estándares, metodologías y herramientas para la gestión y ejecución del mismo y, por último, se prepara y establece un plan para su implementación adecuada y oportuna, incluyendo hitos y revisiones.

El plan de gestión del proyecto software que conducirá el desarrollo se produce como culminación de este proceso.

Actividades a realizar

- Establecer el mapa de actividades para el modelo del ciclo de vida seleccionado.
- Asignar los recursos del proyecto
- Definir el entorno del proyecto
- Planificar la gestión del proyecto

Documentos de salida

- Plan de gestión de proyecto
- Plan de retiro

Técnicas a utilizar:

- Análisis de camino crítico (CPM)
- Análisis Pert
- Diagrama de Gantt
- Técnicas estadística
- Técnicas de simulación (Método de *Montecarlo*)
- Puntos de función
- Modelos empíricos de estimación (COCOMO, Putman)

- Técnicas de descomposición para estimación

Proceso de seguimiento y control del proyecto

Es un proceso iterativo de seguimiento, registro y gestión de costos, problema y rendimiento de un proyecto durante su ciclo de vida. En este proceso se realiza un análisis de riesgos (técnico, económico, operativo y de soporte, y del programa y calendario) que permite identificar los problemas potenciales, determinar su probabilidad de ocurrencia y su impacto, como así también, establecer los pasos para su gestión. Los riesgos identificados y su gestión se documentan en el **plan de contingencias**.

El progreso de un proyecto se revisa y mide con respecto a los hitos establecidos en el plan de gestión del proyecto software.

Actividades a realizar

- Analizar riesgos
- Realizar la planificación de contingencias
- Gestionar el proyecto
- Archivar registros
- Implementar el sistema de informes de problemas

Documentos de salida

- Análisis de riesgos
- Plan de contingencias
- Registro histórico de proyectos

Técnicas a utilizar

- Análisis de riesgo técnico
    - Modelado y simulación estática y dinámica
    - Prototipado
    - Revisiones
    - Auditorías
- Análisis de riesgo económico
    - Análisis de finanzas
    - Retorno de la inversión
- Análisis de riesgo operativo y de soporte
- Análisis de reisgo del programa
    - Análisis de camino crítico (CPM)
    - Técnicas de nivelación de recursos

Proceso de gestión de la calidad del software

Su objetivo es la planificación y administración de las acciones necesarias para proveer una confianza adecuada en la calidad de los productos software, es decir, que satisfagan los requisitos técnicos establecidos.

El proceso de gestión de la calidad del software se documenta en un **Plan de Garantía de la Calidad del Software.** Sus actividades abarcan el ciclo de vida completo del software. Para abordar este proceso de protección del software, con una visión global, se consideran los siguiente tres aspectos principales:

- Métricas del software para el control del proyecto.
- Verificación y validación, incluyendo pruebas y procesos de revisión.
- Gestión de las Configuración del producto software

Las **métricas del software** se definen como la aplicación continua de técnicas basadas en las medidas de los procesos de desarrollo software y sus productos que produzcan información de gestión definitiva y a tiempo, a la vez que se mejoran, los procesos y sus productos.

La verificación y validación del software, involucra actividades imprescindibles para el control de la calidad del software. Se entiende por **verificación** al conjunto de actividades para la comprobación de que un producto software está técnicamente bien construido; es decir, que el producto funciona (¿está el producto correctamente construido?).

En general, comprobar si los productos construidos en una fase del ciclo de vida satisfacen los requisitos establecidos en la fase anterior, diciendo si el producto, hasta el momento, es consistente y completo. De modo complementario la **validación** trata la comprobación de que el producto software construido es el que se deseaba construir; es decir, que el producto funciona como el usuario quiere y hace las funciones que se establecieron (¿el producto construido es el correcto?). En general, comprobar si el software construido satisface los requisitos del usuario. Evidentemente, sólo tiene objeto validar el producto que está verificado (no interesa comprobar que el producto que no funciona es lo que se pedía).

Actividades a realizar

- Planificar la garantía de la calidad del Software
- Desarrollar métricas de calidad
- Gestionar la calidad del software
- Identificar necesidades de mejora de la calidad

Documentos de salida

- Plan de Garantía de Calidad del Software
- Recomendaciones de mejora de Calidad Software

Técnicas a aplicar

- Técnicas de planificación y estimación

- Métricas de Calidad del Software

Proceso de pre-desarrollo

Son los procesos que se deben realizar antes de que comience el desarrollo propiamente dicho del software. Es esfuerzo de desarrollo se inicia con la identificación de una necesidad de automatización. Esta necesidad, para ser satisfecha, puede requerir una nueva aplicación, o un cambio de todo o parte de una aplicación existente. El pre-desarrollo abarca desde el reconocimiento del problema hasta la determinación de los requisitos funcionales a nivel de sistema, pasando por el estudio de la viabilidad de su solución automatizada.

Proceso de exploración de conceptos

Este proceso incluye la identificación de una necesidad, la formulación de soluciones potenciales, su evaluación (Estudio de Viabilidad) y refinamiento a nivel de sistema. Una vez establecido sus límites, se genera el **Informe de la necesidad** del sistema a desarrollar. Este informe, inicia el proceso de asignación del sistema y/o el proceso de requisitos y alimenta los procesos de gestión del proyecto.

El Informe de la Necesidad es un documento que constituye la base de todo el trabajo de ingeniería posterior.

Actividades a realizar

- Identificar ideas o necesidades
- Formular soluciones potenciales
- Conducir estudios de viabilidad
- Planificar la transición del sistema (si se aplica)
- Refinar y finalizar la idea o necesidad

Documentos de salida

- Modelo de la Situación actual
- Modelo del dominio del problema
- Informe preliminar de necesidades
- Soluciones alternativas posibles
- Soluciones recomendadas
- Plan de transición
- Informe del impacto de la transición

Técnicas a usar

- Técnicas de adquisición de conocimientos

- Análisis económico (costo/beneficio)
- Análisis técnico
- Análisis alternativo
- Técnicas de modelado
    - Diagrama de flujos de datos (DFD)
- Prototipado

Proceso de asignación del sistema

Este proceso se realiza cuando el sistema requiere tanto del desarrollo de hardware como de software, o cuando no se puede asegurar que sólo se necesita desarrollo de software. El informe de la necesidad se analiza para identificar las entradas, el procesamiento que se aplica a la entrada, las salidas requeridas y las funciones del sistema total, que permite desarrollar la arquitectura del sistema e identificar las funciones del hardware, del software y de las interfaces. Este proceso culmina con la **especificación de requisitos del software,** la **especificación de requisitos del hardware** y la **especificación de la interface del sistema.**

Debido a que el software es parte de un sistema mayor, se comienza estableciendo los requisitos de todos los elementos del sistema y luego asignando algún subconjunto de estos requisitos al software, para su análisis y refinamiento en el proceso de requisitos. Este planteamiento del sistema es esencial cuando el software debe interrelacionarse con otros elementos, tales con hardware, personas y bases de datos. Antes de la asignación del sistema, el análisis abarca los requisitos globales a nivel de sistema, con una pequeña cantidad de análisis y de diseño a un nivel superior.

El análisis de sistema requiere de una comunicación intensa entre el cliente y el analista. El cliente debe comprender los objetivos del sistema y ser capaz de exponerlos claramente. El analista debe saber qué preguntas hacer, qué consejos dar, qué investigación realizar. Si la comunicación se rompe, el éxito del proyecto entero estará en peligro.

En el análisis de sistemas (proceso de exploración de conceptos), se definen los objetivos del mismo, la información que se va a obtener, la información que se va a suministrar, las funciones, el comportamiento y el rendimiento requeridos. El Analista se asegura en distinguir entre los que necesita el cliente (elementos críticos para la realización), y lo que el cliente "quiere" (elementos deseables pero no esenciales). Una vez que la función, el rendimiento y las interfaces (determinados por el comportamiento) están delimitados, se procede a realizar la tarea denominada asignación. Durante la asignación las funciones son asignadas a uno o más elementos genéricos del sistema, es decir, software, hardware, personal, base de datos, documentación, procedimientos. A menudo, se proponen y evalúan varias asignaciones. Esencialmente, lo que se hace es asignar a cada elemento del sistema, un ámbito de funcionamiento y de rendimiento.

Asignadas las funciones del sistema informático, se puede crear un modelo que represente las interrelaciones entre los distintos elementos del sistema y establezca una base para los posteriores pasos de análisis de requisitos y de diseño. Se representa el sis-

tema definido mediante modelos de la arquitectura del sistema (salida, entrada, proceso y control, interfaz de usuario y mantenimiento y autocomprobación). En primer lugar, se realiza un **Diagrama de contexto** de la arquitectura (DC), que establece los límites de información entre los que se está implementando el sistema y el entorno en el que va a funcionar. En esencia el DC ubica el sistema en el contexto de su entorno externo.

Se refina el diagrama de contexto de la arquitectura, considerando con más detalle la función del sistema. Se identifican los subsistemas principales que permiten el funcionamiento del sistema considerado en el contexto definido por el DC. El diagrama de flujo de la arquitectura (DF) muestra los subsistemas importantes y las líneas importantes de flujo de información (control y datos). En esta etapa, cada uno de los subsistemas pueden contener uno o más elementos del sistema (por ejemplo: hardware, software, personal), según hayan sido asignados.

El diagrama inicial de flujo de la arquitectura, se constituye en el nodo raíz de la jerarquía de DF. Se puede ampliar cada subsistema del DF inicial en otro diagrama de arquitectura, dedicado exclusivamente a él. Este proceso de descomposición de arriba abajo (top-down) permite disponer de una jerarquía de DF, donde cada uno de los DF del sistema se puede utilizar como punto de partida para los posteriores pasos de ingeniería del subsistema que describe.

Actividades a realizar

- Analizar las funciones del sistema
- Desarrollar la arquitectura del sistema
- Descomponer los requisitos del sistema

Documentos de salida

- Especificación de requisitos funcionales del software
- Especificación de requisitos funcionales del software
- Especificación de la interfaz del sistema
- Descripción funcional del sistema
- Arquitectura del sistema

Técnicas a utilizar

- Técnicas de adquisición de conocimiento
- Técnicas de modelado
    - Diagrama de flujo de datos (DFD)

Proceso de desarrollo

Son procesos que se deben realizar para la construcción del producto software. Éstos definirán qué información obtener y cómo estructura los daros, qué algoritmos usa para procesar los datos y cómo implementarlos y qué interfaces desarrollar para operar con el software y cómo hacerlo.

A partir del informe de la necesidad, con el soporte de las actividades de los procesos integrales y bajo el plan de gestión del proyecto, los procesos de desarrollo producen el software (código y documentación).

Proceso de requisitos

Incluye las actividades iterativas dirigidas al desarrollo de la especificación de requisitos software. Para la determinación completa y consistente de los requisitos del software el análisis se enfatiza sobre la salida resultante, la descomposición de datos, el procesamiento de los datos, las bases de datos (si existen) y los interfaces del usuario, del software y del hardware.

La **especificación de requisitos del software** es el establecimiento conciso y preciso de un conjunto de requisitos que deben ser satisfechos por un producto software, indicando, siempre que sea apropiado, el procedimiento mediante el cual se puede determinar si se satisfacen los requisitos dardos. Describe los requisitos funcionales, de rendimiento, y de interfaz del software y definir los entornos de operación y de soporte. Este documento es la salida con que culmina este proceso.

Un requisito es una condición o característica que debe tener o cumplir un sistema o componente de un sistema para satisfacer un contrato, norma, especificación, u otro documento formalmente impuesto. El conjunto de todos los requisitos forma la base para el desarrollo consiguiente del sistema o componente del sistema.

Existen tres tipos de requisitos: funcionales, de rendimiento y de interfaz. Un **requisito funcional** especifica la función que un sistema o componente de un sistema debe ser capaz de realizar (por ejemplo: emitir facturas. Un **requisito de rendimiento** especifica una característica numérica tanto estática (por ejemplo: números de usuarios simultáneos, número de archivos y volumen del os mismos, etc.), como dinámica (por ejemplo: el 95% de las transacciones se deben procesar en menos de un segundo) que debe tener un sistema o componente de un sistema. Los **requisitos de interfaz** determinan las características que el software debe soportar para cada interfaz humano del producto software (interfaz del usuario), las características lógicas de cada interfaz del producto software y las componentes hardware del sistema (interfaz de hardware), el uso de ortos productos software (por ejemplo: un sistema de gestión de base de datos, un sistema operativo, o un paquete matemático) e interfaces con otros sistemas de aplicación (interfaz de software). Por ejemplo: que la interfaz del usuario cuente con ventanas superpuestas.

Existen además algunos atributos del software (seguridad, consistencia, facilidad de traza, etc.) que pueden dar lugar a requisitos específicos del mismo, por ejemplo; que el acceso a ciertos datos sea restringido.

Actividades a realizar

- Definir y desarrollar los requisitos del software
- Definir los requisitos de interfaz
- Priorizar e integrar los requisitos del software

Documentos de salida

- Especificación de requisitos del software.
- Requisitos de interfaz con el usuario
- Requisitos de interfaz con otro software.
- Requisitos de interfaz con el hardware.
- Requisitos de interfaz con el sistema físico.

Técnicas a utilizar

Técnicas orientadas a los procesos

- Análisis estructurado
  - Diagramas de flujo de datos (DFD)
  - Diccionario de datos (DD)
  - Especificación de procesos primitivos (EPP)
- SADT (Structured Analysis and Design Techniques)
  - Diagramas de transición de estados
  - Diagramas de descomposición
    - ✓ WRS (Working Breakdown Structure)
    - ✓ RBS (Resources Breakdown Structure)
    - ✓ OBS (Object Breakdown Structure)
- Antigramas (Diagramas de actividades)

Técnicas orientadas a los datos

- Diagramas entidad-relación
- Datagramas (Diagramas de datos)

Técnicas orientadas a los objetos:

- Diagrama de Clases/Objetos
- Jerarquías de Clases/Objetos

Técnicas formales de especificación:

- Técnicas relacionales
  - Ecuaciones implícitas
  - Relaciones recurrentes
  - Axiomas algebraicos
  - Expresiones regulares
- Técnicas orientadas al estado
  - Tablas de decisión
  - Tablas de eventos
  - Tablas de transición
  - Mecanismos de estados finitos
  - Redes de Petri

- Técnicas de prototipado

Proceso de diseño

Es el proceso centra que unifica los procesos de desarrollo y de mantenimiento del software. Su objetivo es desarrollar una representación coherente y organizada del sistema software que satisfaga la especificación de requisitos del software. La calidad de dicha representación se puede evaluar. El proceso de diseño traduce el "qué hacer" de las especificaciones de los requisitos en el "cómo hacerlo" de las especificaciones diseño. Inicialmente, le representación describe una visión sistemática y holística del software. Posteriores, refinamientos de diseño conducen a una representación que se acerca al código fuente.

El diseño del software puede verse desde los perspectivas: la técnica y la de gestión del proyecto. Desde el punto de vista técnico, el diseño comprende cuatro actividades: diseño de datos, diseño arquitectónico, diseño procedimental y diseño de los interfaces. Desde el punto de vista de gestión del proyecto, el diseño va del diseño arquitectónico (diseño preliminar o diseño de alto nivel) al diseño detallado (diseño de bajo nivel).

Desde el punto de vista de gestión del proyecto, el nivel de diseño arquitectónico o preliminar se centra en las funciones y estructuras de las componentes que conforman el sistema de software. El nivel de diseño detallado se ocupa del refinamiento de las representaciones algorítmicas que se usan para cada componente modular del software.

El proceso del diseño comienza con la actividad de realizar el diseño arquitectónico. Esta actividad genera la **descripción de diseño arquitectónico del software** en donde se describe el diseño de cada una de las componentes software, se especifican los datos, las relaciones y las restricciones y se definen todos los interfaces externos (usuario, software y hardware) y los interfaces internos (entre las componentes).

La última actividad del proceso de diseño es realizar el diseño detallado, donde se genera la **descripción de diseño del software (DDS)** que especifica la estructura de los datos, los algoritmos y la información de control de cada componente software, y los detalles de los interfaces (usuarios, software y hardware).

El diseño detallado deriva del diseño preliminar, en consecuencia sus correspondientes actividades se realizan en secuencia mientras que el resto de las actividades de este proceso (analizar el flujo de información, diseñar la base de datos, diseñar los interfaces, seleccionar o desarrollar algoritmos) se ejecutan en paralelo. Estas últimas producen refinamientos del diseño que son también entradas a la actividad de diseño detallado.

Una actividad relevante es la del diseño de las base de datos. Esta comprende el diseño conceptual, lógico y físico de la base de datos. Los requisitos se modelan dentro de un esquema externos que describe las entidades de datos, atributos, relaciones y restricciones. Los distintos esquemas externos se integran en un esquema conceptual único. EL esquema conceptual se *aplica* entonces en un esquema lógico dependiente de la implementación. Finalmente, se definen las estructuras físicas de datos y los caminos de acceso. El resultado de esta actividad es la descripción de la base de datos.

Como ya dijimos anteriormente, desde el punto de vista técnico y en el contexto de los diseños preliminar y detallado, se llevan a cabo varias actividades de diseño diferentes: diseño de datos, diseño arquitectónico, diseño procedimental y diseño de la interfaz.

El diseño de datos transforma el modelo del campo de información, creado durante el análisis, el las estructuras de datos que se van a requerir para implementar el software. El diseño arquitectónico define las relaciones entre los principales elementos estructurales del programa. EL objetivo principal de este diseño es desarrollar una estructura de programa modular y representar las relaciones de control entre los módulos. Además, el diseño arquitectónico define las relaciones entre los principales elementos estructurales del programa. El objetivo principal de este diseño es desarrollar una estructura de programa modular y representar las relaciones de control entre los módulos. Además, el diseño arquitectónico mezcla la estructura de programas y la estructura de datos define las interfaces que facilitan el flujo de los datos a lo largo del programa. El diseño procedimental transforma los elementos estructurales en una descripción procedimental del software; se realiza después de que se ha establecido la estructura del programa y de los datos. El diseño del interfaz establece principalmente la disposición y los mecanismos para la interacción hombre-máquina.

El diseño es el proceso donde se asiente la calidad del desarrollo. El diseño produce las representaciones del software, de las que puede evaluarse su calidad. El diseño es la única forma mediante la cual se puede traducir con precisión los requisitos del cliente en un producto o sistema acabado. El diseño del software sirve como base de todas las posteriores etapas del desarrollo y del proceso de mantenimiento. Sin diseño, se puede construir un sistema inestable, un sistema que falle cuando se realicen pequeños cambios; un sistema que pueda ser difícil de probar, un sistema cuya calidad no pueda ser evaluada hasta más adelante en el proceso de Ingeniería de Software, cuando quede poco tiempo y se haya gastado ya mucho dinero.

El proceso de diseño en la Ingeniería de Software conduce a un buen diseño mediante la aplicación de principios fundamentales de diseño (abstracción, refinamiento sucesivo, modularidad, estructura jerárquica de los módulos, estructura de los datos, jerarquía de control, procedimiento realizado por capas funcionales, ocultamiento de información), de métodos sistemáticos y de una adecuada revisión.

Para evaluar la calidad de una representación del diseño, se deben tener en cuenta, entre otros, los siguientes criterios de calidad del diseño:

1. Un diseño debe exhibir una organización jerárquica que haga un uso inteligente del control entre los componentes del software.
2. Un diseño debe ser modular, es decir que el software debe estar dividido de forma lógica en elementos que realicen funciones y sub-funciones específicas.
3. Debe contener representaciones distintas y separadas de los datos y los procedimientos.
4. Debe llevar a módulos (por ejemplo: subrutinas o procedimientos) que exhiban características funcionales diferentes.
5. Debe llevar a interfaces que reduzcan la complejidad de las conexiones entre los módulos y el entorno exterior.

6. Debe obtenerse mediante un método que sea reproducible y que esté conducido por la información obtenida durante el análisis de los requisitos del software.

La modularidad se ha convertido en un enfoque ampliamente aceptado, ya que un diseño modular reduce la complejidad, facilita los cambios (un aspecto crítico de la facilidad de mantenimiento del software) y produce como resultado, una implementación más sencilla permitiendo el desarrollo paralelo de las diferentes partes de un sistema.

En la arquitectura del software, para la definición de módulos, se utiliza los conceptos de abstracción y de ocultamiento de información que derivan en el concepto de independencia funcional.

La independencia funcional se logra desarrollando módulos con una función definida y específica (máxima cohesión) y una aversión a una excesiva interacción con otros módulos (mínimo acoplamiento). Se trata de diseñar software de forma que cada módulo se centre en una subfunción específica de los requisitos y tenga una interfaz sencilla, cuando se ve desde otra parte de la estructura del software.

Así, la independencia se mide con dos criterios cualitativos: la cohesión y el acoplamiento. La cohesión es una medida de la fortaleza funcional relativa de algún módulo. El acoplamiento es una medida de la interdependencia relativa entre los módulos de una estructura de programa.

La notación de diseño, junto con los conceptos de la programación estructurada, permite al diseñador, representar los detalles procedimentales, facilitando su traducción al código, que puede realizarse automáticamente a través de una herramienta CASE (Computer Aided Software Engineering).

Actividades a realizar:

- Realizar el diseño arquitectónico
- Analizar el flujo de información
- Diseñar la base de datos (si se aplica)
- Diseñar las interfaces
- Seleccionar y desarrollar algoritmos (si se aplica)
- Realizar el diseño detallado

Documentos de salida

- Descripción del diseño del software
- Descripción de la arquitectura del software
- Descripción del flujo de información
- Descripción de la base de datos
- Descripción de las interfaces
- Descripción de los algoritmos.

Técnicas a utilizar:

Técnicas orientadas a los procesos

- Diseño estructurado

    - Análisis de transformación
    - Análisis de transacción
- Diseño del diálogo de las interfaces
- Diseño lógico o del perfil
- HIPO (*Hierarchy Input Process Output)*

Técnicas orientadas a los datos

- Modelo lógico de datos
- Modelo físico de datos

Técnicas orientadas a los objetos

- Modelo de clases/objetos
- Diagrama de módulos

Técnicas de diseño de bajo nivel

- Programación estructurada
    - Diagramas arborescentes
    - Diagramas de Chapín
- Programación orientada a objetos
    - Diagrama de procesos
- Técnicas de prototipado
- Técnicas de refinamiento

Proceso de implementación

Este proceso transforma la representación del diseño detallado de un producto software a una realización en un lenguaje de programación apropiada. El proceso de implementación produce el código fuente, es código de la base de datos (de ser necesario) y la documentación que constituyen la manifestación física del diseño de acuerdo de los estándares y metodologías del proyecto. Además, en este proceso se debe integrar el código y la base de datos. En el caso en el que el sistema incluya componentes de hardware y de software, se debe planificar y realizar la integración del sistema.

La salida de este proceso está sujeta a las pruebas de verificación y validación adecuadas. El código y la base de datos, junto con la documentación producida durante los procesos previos son la primera representación completa del producto software.

Actividades a realizar

- Crear los datos de prueba
- Crear el código fuente
- Generar el código objeto

- Crear la documentación de operación
- Planificar la integración
- Realizar la integración

Documentos de salida

- Datos para las pruebas
- Documentación del sistema
- Documentación del usuario
- Plan de integración
- Sistemas software integrado

Técnicas a utilizar

- Warnier
- Jackson
- Lenguajes de programación

Proceso de post-desarrollo

Son los procesos que se deben realizar para instalar, operar, soportar, mantener y retirar un producto software. Se realizan luego de la construcción del software. Se aplican a las últimas fases del ciclo de vida del software.

Una vez terminada la prueba del software, este está casi preparado para ser entregado a los usuarios finales. Sin embargo, antes de la entrega, se llevan a cabo una serie de actividades de garantía de calidad para asegurar que se han generado y catalogado los registros y documentos internos adecuados, que se ha desarrollado una documentación de alta calidad para el usuario y que se han establecido los mecanismos apropiados de control de configuraciones. Entonces, el software ya puede ser distribuido a los usuarios finales.

Tan pronto como se entregue el software a los usuarios finales, el trabajo del Ingeniero de Software cambia. En ese momento, el enfoque pasa de la construcción al mantenimiento, corrección de errores, adaptación al entorno y mejora dela funcionalidad.

En todos los casos, la modificación del software, no sólo afecta al software sino también a la configuración del software. Es decir, todos los documentos, datos y programas desarrollados en la fase de planificación y desarrollo.

Proceso de instalación

Implica el transporte y la instalación de una sistema software desde el entorno de desarrollo al entorno de destino. Incluye la carga, de ser necesario, de la base de datos, las modificaciones necesarias del software, las comprobaciones en el entorno de destino y la aceptación del cliente. Si durante la instalación surge un problema, se identifica e informa acerca de él.

El proceso de instalación verifica que se implemente la configuración adecuada del software y culmina con la aceptación formal del mismo por parte del cliente, conforme a lo especificado en el plan de gestión del proceso software y la realización con éxito de la prueba de aceptación del usuario.

Actividades a realizar

- Planificar la instalación
- Distribuir el software
- Instalar el software
- Cargar la base de datos (de ser necesario)
- Aceptar el software en el entorno de operación
- Realizar las actualizaciones (instalar el software probado)

Documentos de salida

- Plan de instalación del software
- Informe de instalación

Proceso de operación y soporte

Involucra la operación del sistema por parte del usuario y el soporte continuo al usuario que incluye asistencia técnica, consultas con el usuario y registro de las peticiones de soporte en el **histórico de peticiones de soporte.** Así, este proceso puede desencadenar la actividad del proceso de mantenimiento que provee información re-entrante al ciclo del vida del software.

Actividades a realizar

- Operar el sistema
- Proveer de asistencia técnica y consultas
- Mantener el histórico de peticiones de soporte

Documentos de salida

- Histórico de peticiones de soporte

Proceso de Mantenimiento

Se interesa por los errores, defectos, fallos, mejoras y cambios del software. Un requisito de mantenimiento del software inicia los cambios del ciclo de vida del software. Este se reasigna y ejecuta.

El mantenimiento se centra en el cambio que va asociado a la corrección de errores, a las adaptaciones requeridas por la evolución del entorno del software y a las modificaciones del software debidas a los cambios de los requisitos del cliente, dirigidos a reforzar o ampliar el sistemas. El proceso de mantenimiento vuelve a aplicar los pasos del ciclo de

vida, pero en el contexto del software ya existente; de este modo se considera el proceso de mantenimiento como iteraciones de desarrollo. Durante el mantenimiento se encuentran tres tipos de cambios:

- Corrección: Incluso llevando a cabo las mejores actividades de garantía de calidad, es muy probable que el cliente descubra defectos en el software. El mantenimiento correctivo cambia el software para corregir los defectos.
- Adaptación: Con el paso del tiempo es probable que cambie el entorno original, por ejemplo, la CPU, el sistema operativo, periféricos), para el que se desarrolló el software. El mantenimiento adaptativo, consiste en modificar el software para acomodarlo a los cambios de su entorno externo.
- Mejora: Conforme utilice el software el cliente/usuario puede descubrir funciones adicionales que podría interesar que estuvieran incorporadas en el software. El mantenimiento perfectivo amplía el software más allá de sus requisitos funcionales originales.

La salida de este proceso son **recomendaciones de mantenimiento** que entran al ciclo de vida del software en el proceso de exploración de conceptos para mejorar la calidad del sistema software.

Actividades a realizar

- Reaplicar el ciclo de vida del software

Documentos de salida

- Orden de mantenimiento
- Recomendaciones de mantenimiento

Proceso de retiro

Es la jubilación de un sistema existente, su soporte activo o de su uso mediante el cese de su operación o soporte o mediante su reemplazo tanto por un nuevo sistema como una versión actualizada del sistema existente. Si el sistema en uso, sea manual o automatizado, se reemplaza por un nuevo sistema que requiere un período de operación dual, denominado ensayo en paralelo. En este período se utiliza el sistema en retiro para los resultados oficiales, mientras se completa la preparación del nuevo sistema para la operación formal. Es un período de formación del usuario sobre el nuevo sistema y de validación del mismo.

Actividades a realizar

- Notificar al usuario
- Conducir operaciones en paralelo (si se aplica)
- Retirar el sistema

Documentos de salida

- Plan de retiro

Procesos integrales del proyecto

Son procesos simultarios y complementarios a los procesos orientados al desarrollo. Incluyen actividades imprescindibles para que el sistema construido sea fiable (proceso de verificación y validación (Gestión de configuración) y sea utilizado al máximo de sus capacidades (procesos de formación, documentación).

Los procesos integrales comprenden dos tipos de actividades:

a) Aquella que se realizan discretamente y se aplican dentro de un ciclo de vida del software y …
b) Aquellas que se realizan para completar otra actividad

Estas son actividades que se invocan (llamadas como una subrutina), y no se aplican al modelo del ciclo de vida del software para cada instancia.

Proceso de verificación y validación

Abarca la planificación y la realización de todas las tareas de verificación. Incluye pruebas de verificación, revisiones y auditorías. También las taras de validación, que se ejecutan durante el ciclo de vida del software para asegurar que se satisfacen todos los requisitos del software.

Una actividad útil para la verificación y la validación del software es la prueba del software. Constituyen el proceso de ejecución del software con determinados datos de entrada, denominados juegos de ensayo, para observar los resultados que produce y compararlos con los resultados que teóricamente (de acuerdo con las especificaciones), debería producir, para esos datos de entrada, con el objeto de detectar posibles fallos. Las pruebas del software solo podrán realizarse cuando en el proceso de desarrollo ya existía código ejecutable.

La depuración es un proceso frecuentemente asociado a las pruebas y a algunas pruebas (y algunas otras actividades de verificación y validación) que consiste en tratar de deducir dónde están los defectos en el software que provocan que éste no funcione correctamente. Estudia los resultados de pruebas y otras actividades de control para intentar buscar qué está mal en el software.

Este proceso se aplica en cada procesos y producto del ciclo de vida del software.

Actividades a realizar

- Planificar la verificación y validación
- Ejecutar las tareas de verificación y validación
- Recoger y analizar los datos de las métricas
- Planificar las pruebas
- Desarrollar las especificaciones de las pruebas
- Ejecutar las pruebas

Documentos de salida

- Plan de verificación y validación
- Informe de evaluación
- Plan de pruebas
- Especificación de pruebas
- Informe resumen de las pruebas
- Software probado

Técnicas a utilizar

- Técnicas de prueba de caja blanca
  - Cobertura de sentencias
  - Cobertura de decisión o de ramificación
  - Cobertura de condición
  - Cobertura de decisión / condición
  - Cobertura de condición múltiple
- Técnicas de prueba de caja negra
  - Partición de equivalencias
  - Análisis de valores límites
  - Gráficos de causa-efecto
  - Conjetura de errores
- Revisiones formales
- Auditorías

Proceso de gestión de la configuración

Este proceso involucra un conjunto de actividades desarrollada para gestionar los cambios durante todo el ciclo de vida del software. Identifica la estructura de un sistema (qué rutinas, módulos, datos, archivos, etc, lo componen) en un momento dado, aun cuando se está desarrollando a lo que se denomina **configuración del sistema.** Su objetivo es el control de los cambios en el sistema, mantener su coherencia y su "rastreabilidad" o "trazabilidad", y poder realizar auditorías de control sobre la evolución de las configuraciones.

La gestión de la Configuración, realiza las siguientes funciones:

a) Identificación de la configuración del sistema: descripción documentada de las características reales del sistema en un determinado momento.
b) Control de la configuración: Establece la configuración inicial o básica y controla los cambios en los elementos de la misma.
c) Informe del estado de la configuración, y

d) Auditorías de configuración: revisiones independientes de la configuración para comprobar que los elementos de la configuración cumplen los requisitos de configuración establecidos.

En resumen, la gestión de configuración del software, identifica los elementos de un proyecto de desarrollo del software (especificaciones, código, planes, etc) y provee tanto el control de los elementos identificados como la generación de informes de estado de la configuración. Todo ésto con el objeto de conseguir visibilidad en la gestión y responsabilidad de la misma, durante el ciclo de vida del software.

Actividades a realizar

- Planificar la gestión de la configuración
- Realizar la identificación de la configuración
- Realizar el control de la configuración
- Realizar la información del estado de la configuración

Documentos de salida

- Plan de gestión de la configuración del software
- Orden de cambio de ingeniería
- Cambio de estado
- Informe de estado

Proceso de desarrollo de documentación

El proceso de desarrollo de documentación para el desarrollo y uso del software es el conjunto de actividades que planifican, diseñan, implementan, editan, producen, distribuyen y mantienen los documentos necesarios para los desarrolladores y los usuarios.

Actividades a realizar

- Planificar la documentación
- Implementar la documentación
- Producir y distribuir la documentación

Documentos de salida

- Plan de documentación

Proceso de formación

Incluye la planificación, desarrollo, validación e implementación de los programas de formación de desarrolladores, personal de soporte técnico y clientes y la elaboración de materiales de formación adecuados.

Para conseguir una utilización efectiva del sistema software se debe proporcionar a los usuarios del sistema, instrucciones, guía y ayuda para el entendimiento de las capacidades del sistema y de sus limitaciones. Por lo que es imprescindible la formación de los usuarios en el nuevo sistema.

El desarrollo de productos software de calidad depende en gran medida de los conocimientos de las personas y del personal especializado involucrado en el proyecto. Es esencial la formación de los desarrolladores y personal de soporte técnico.

Actividades a realizar

- Planificar el programa de formación
- Desarrollar los materiales de formación
- Validar el programa de formación
- Implementar el programa de formación

Documentos de salida

- Plan de formación

Mapa de Actividades

La decisión de qué ciclo de vida se elegirá para el proyecto en cuestión. Una vez que se ha hecho tal selección, y guiado en cierto modo por ella, se debe adaptar el proceso software genérico al modelo de ciclo de vida elegido. Es decir, establecer el mapa de actividades del proyecto. El proceso software visto en el apartado anterior es un proceso genérico, que obligatoriamente debe adaptarse para cada proceso. La adaptación se lleva a cabo precisamente mediante el establecimiento del mapa de actividades.

El mapa de actividades es una tabla donde se marcan qué actividades del proceso software genérico se van a ejecutar para un determinado proyecto. Existen desarrollados ya ciertos mapas de actividades para los tipos de proyectos más comunes (proyecto grande, proyecto pequeño, etc.), que pretenden facilitar la labor del jefe de proyecto. En cualquier caso, las tablas existentes se deben siempre comprobar y adaptar para un proyecto concreto. Hay que insistir en que no hay dos proyectos de desarrollo de software iguales.

El tipo de proyecto está muy estrechamente relacionado con el ciclo de vida. Por ejemplo, un proyecto pequeño casi siempre se corresponde con un ciclo de vida en cascada; un proyecto medio innovador, se suele corresponder con un ciclo de vida con prototipado; etc.

Por tanto, como puede verse en la siguiente figura**,** el mapa de actividades es una tabla de dos entradas. Una entrada es el proceso software con sus actividades y la otra entrada, el ciclo de vida elegido descompuesto en sus etapas.

Entonces, el mapa de actividades es una tabla de doble entrada, donde una entrada es el proceso software con sus actividades y la otra las etapas correspondientes al ciclo de vida elegido:

Ciclo de vida descompuesto en etapas

Proceso de Software		**Requisitos**	**Diseño**	**...**
	Actividad 1			
	Actividad 2			
	...			
	...			
	...			
	Actividad n			

En el mapa de actividades simplemente se marcan con una cruz las actividades que se llevarán a cabo. Además, puede incluirse más información del tipo: importancia de la actividad, si está marcada con un "**+**" significa mucha y si está marcada con un "—" significa normal; tipo de actividad, obligatoria (marcada con una **O***)* o condicional (marcada con una **C**).

A continuación, en la **tabla** 2, aparece un ejemplo de mapa de actividades para un proyecto que ha elegido un ciclo de vida en cascada con nueve etapas definidas del siguiente modo:

Factibilidad (FA). Definir el producto software, y determinar su factibilidad en el ciclo de vida y superioridad con respecto a productos alternativos.

Requisitos (RS). Una completa especificación validada de las funciones requeridas, interfaces, y rendimiento para el producto software.

Diseño del producto (DP). Una completa especificación verificada de la arquitectura del hardware-software, estructura de control, y estructura de datos para el producto, junto con otros componentes necesarios como los manuales del usuario preliminar y los planes de prueba.

Diseño detallado (DD). Especificación verificada de la estructura de control, estructura de datos, relaciones de interfaz, tamaño, algoritmos claves y suposiciones de cada componente de programa.

Codificación (CO). Un completo conjunto verificado de componentes del programa.

Integración (IN). Un adecuado funcionamiento del producto software compuesto de los componentes software.

Implementación (IM). Un sistema hardware-software funcionando operacionalmente a pleno, incluyendo tales objetivos como conversión del programa y datos, instalación, y entrenamiento.

Operación y mantenimiento (OM). Un funcionamiento actualizado del sistema hardware-software. Este sub-objetivo se repite para cada actualización.

Retiro (RE). Una transición adecuada de las funciones realizadas para el producto y sus sucesores (si existe).

ACTIVIDADES	FA	RS	DP	DD	CO	IN	IM	OM	RE
Proceso de selección de un MCVS									
Identificar los posibles MCVS	X								
Seleccionar un modelo para el proyecto	X								
Proceso de iniciación, planificación y estimación del proyecto									
Establecer la matriz de actividades	X								
Asignar recursos del proyecto	X	X	X	X	X	X	X	X	X
Definir el entorno del proyecto	X	X		X					
Planificar la gestión del proyecto	X	X							
Proceso de seguimiento y control del proyecto									
Analizar los riesgos	X	X	X	X	X	X	X		
Realizar la planificación de contingencia		X	X	X	X	X	X		
Gestionar el proyecto	X	X	X	X	X	X	X	X	X
Implementar el sistema de informes de problemas	X	X	X	X	X	X	X	X	X
Archivar registros		X	X	X	X	X	X	X	X
Proceso de gestión de calidad del software									
Planificar la garantía de calidad del software		X	X						
Desarrollar las métricas de calidad		X	X	X					
Gestionar la calidad del software	X	X	X	X	X	X	X	X	X
Identificar necesidades de mejora de la calidad	X	X	X	X	X	X	X	X	X
Proceso de exploración de conceptos									
Identificar las ideas o necesidades	X								
Formular las soluciones potenciales	X	X							

ACTIVIDADES	FA	RS	DP	DD	CO	IN	IM	OM	RE
Dirigir los estudios de viabilidad	X	X							
Planificar la transición del sistema (si aplica)	X	X							X
Refinar y finalizar la idea o necesidad		X							
Proceso de asignación del sistema									
Analizar las funciones del sistema		X	X						
Desarrollar la arquitectura del sistema		X	X						
Descomponer los requisitos del sistema		X							
Proceso de análisis de requisitos									
Definir y desarrollar los requisitos de Sw		X	X						
Definir los requisitos de interfaz		X	X						
Priorizar e integrar los requisitos de Sw		X	X						
Proceso de diseño									
Realizar el diseño preliminar			X						
Analizar el flujo de información			X	X					
Diseñar la base de datos (si aplica)			X	X					
Diseñar las interfaces			X	X					
Seleccionar o desarrollar algoritmos (si aplica)		X		X					
Realizar el diseño detallado				X					
Proceso de implementación e integración									
Crear los datos de prueba			X	X					
Crear el código fuente			X	X					
Generar el código objeto			X	X					
Crear la documentación de operación			X	X					
Planificar la integración			X						
Realizar la integración				X					
Proceso de instalación y aceptación									
Planificar la instalación						X			
Distribuir el software							X		
Instalar el software							X		
Cargar la base de datos (si aplica)							X		
Aceptar el software en el entorno de operación							X		
Realizar las actualizaciones								X	
Proceso de operación y soporte									
Operar el sistema								X	
Proveer de asistencia técnica y consultas								X	
Mantener el histórico de peticiones de soporte								X	
Proceso de mantenimiento									
Realizar el mantenimiento correctivo								X	
Reaplicar el ciclo de vida del software								X	
Proceso de retiro									
Notificar al usuario								X	X
Conducir operaciones en paralelo (si aplica)									X

ACTIVIDADES	FA	RS	DP	DD	CO	IN	IM	OM	RE
Retirar el sistema									X
Proceso de verificación y validación									
Planificar la verificación y validación		X	X						
Ejecutar las tareas de verificación y validación		X	X	X	X	X	X	X	X
Recoger y analizar los datos de las métricas		X	X	X	X	X	X	X	X
Planificar la pruebas				X	X				
Desarrollar las especificaciones de las pruebas				X	X				
Ejecutar las pruebas					X	X	X		
Proceso de configuración									
Planificar la gestión de configuración		X	X						
Realizar la identificación de la configuración		X	X	X	X	X			
Realizar el control de la configuración			X	X	X	X	X	X	X
Realizar la información del estado de la configuración			X	X	X	X	X	X	X
Proceso de documentación									
Planificar la documentación		X	X						
Implementar la documentación				X	X				
Producir y distribuir la documentación						X	X		
Proceso de formación									
Planificar el programa de formación		X	X						
Desarrollar los materiales de formación			X	X	X	X			
Validar el programa de formación						X	X		
Implementar el programa de formación							X		

Como puede verse, el mapa marca qué actividades del proceso software deberán realizarse en cada una de las etapas del ciclo de vida. Existen actividades que es necesario realizarlas una única vez (por ejemplo, la selección de un modelo de ciclo de vida). Sin embargo, existen otras actividades que se realizan en cada una de las etapas (por ejemplo, gestionar el proyecto). Obviamente, dependiendo del ciclo de vida elegido el mapa de actividades variará.

Por lo tanto, el mapa de actividades es el primer paso para conseguir una organización del proyecto que lleve a su gestión. A partir del mapa, puede pasarse a una estimación del tiempo y costo de cada una de las actividades, y por lo tanto, del proyecto global; a una asignación de recursos para cada actividad, etc.